ÖREBRO UNIVERSITET

Teacher: Todor Stoyanov
Phone: 0700738018

First Examination HT 2019

Datorteknik för civilingenjörer

Date: 2019-10-30

Time: 08.15 – 12.15

Material: no supplementary materials allowed

Number of exercises: 5

Total points: 50 (25 points required to pass)

Number of pages: 4 (incl. this page)

Please note:

- Read carefully the instructions before proceeding with the exam.
- Only write on **one side** of the page.
- When handing in, please organize and label pages in the **correct order**.
- Write your **code on all pages, do not** write your **name** anywhere.
- Do **not** use a **red pen or pencil**.
- If the exercise is unclear, make **reasonable assumptions** and write them down.
- Please use **clear handwriting**.
- Answers must be given **in English**. Use **short and complete sentences**.
- Provide **accurate** and **succinct** explanations and motivate your answers.

Good luck!

1 Digital Logic and Representation (5 points)

a: You are given two 8-bit full adders. Devise a circuit that uses the adders to perform addition of two 16-bit numbers. You may, if necessary, use any additional NAND or NOR gates. Motivate your design. (2 point)

b: A special-purpose 8-bit microcontroller needs to control the water level in a tank. The water level is measured as a depth in a unit of meters, and you need to control it with a milimiter precision. Devise an 8-bit floating point representation suitable for this task. What would be the maximum water level you can represent? You can assume that the minimum water level is 0.0 meters. (3 points)

2 Architectures (5 points)

a: Describe the Harvard and Von Neumann architectures and sketch the main components in each. What are the advantages and disadvantages of each architecture? (2 points)

b: You are devising an architecture for processing camera images in real-time. You have three procesors available. The operations you are implementing are:

1: compute the mean pixel value μ (that is the sum of all pixels divided by the number of pixels)

2: compute the variance σ (the average difference between a pixel and the mean)

3: for each pixel, compute a new value as $p_{new} = (p_{old} - \mu)/\sigma$

Argue what would be the advantages and disadvantages of have either a pipelined or a parallel architecture for this task. (3 points)

3 Processors (20 points)

In this exercise we will use a 16-bit processor with 16 general-purpose registers, one accumulator register (*acc*), and one comparison register (*com*). The word-size is 16 bits and the processor has separate instruction and data memories. The processor implements the following instructions:

push Pushes a value from register r_a into the accumulator register *acc*.

pop Pops the value of *acc* into r_a.

add Adds the integers from registers r_a and r_b and store the result in *acc*.

sub Computes $acc = r_a - r_b$.

load Load an integer from data memory into register r_b. The integer is located at data address $r_a + offset$.

store Store an integer from register r_b into the data memory at location $r_a + offset$.

jump Set the program counter to the integer address at location $r_a + offset$ in instruction memory.

cmp Compares the values in r_a and r_b and sets the comparison register *com* to 1 if $r_a > r_b$ and to 0 otherwise.

bne Jumps to location $r_a + offset$, if the value of *com* is equal to 1.

noop No operation, idle.

The following questions all refer to the above architecture

a: Construct a binary representation for each instruction. Describe if you are using a fixed operand number, and if yes, how many operands. Try to optimize your encoding for achieving the largest possible number represented as an offset. What is the largest offset you can encode, assuming that you are using a two's complement representation? (5 points)

b: The instruction set does not allow for a division operation. Assume we want to use this processor to emulate one which provides a division instruction **div** r_a**,** r_b which computes the integer division r_a/r_b for positive numbers r_a, r_b and stores the result in *acc*. Write an assembly code routine to implement this instruction. You can assume that the jump and bne instructions can be called with a label, instead of an explicit offset. (5 points)

Given the following assembly code snippet:

```
        load   r5(12),r1   # r1=memory[r5+12]
label1: cmp    r1,r2       # com=r1>r2
        bne    label2      # conditional jump
        sub    r2,r3       # acc=r2-r3
        pop    r2          # r2=acc
        jmp    label1      # jump to label2
label2: store  r4,r2       # memory[r4]=r2
```

c: Explain how a two stage assembler would process the code above to produce a binary executable program. Illustrate what the binary would look like after each stage pass. Assume that register r_0 always holds the current value of the program counter. (3 points)

d: Your task is to now extend this architecture with an instruction pipeline, in order to improve the execution speed. Explain how a pipeline will help you to get faster cycle times. Use three pipeline stages and explain what each stage would do. (3 points)

e: Insert *noop* statements in the above code to indicate where pipeline stalls would occur. (2 points)

f: Imagine that registers 0 through 7 are located on register bank A, while registers 8 through 15 are on register bank B. Modify the code above to make use of the register bank parallelism. (2 points)

4 Memory (10 points)

a: What is the expected performance gain of using a single cache between processor and memory? (1 point)

b: A Direct Mapped Memory Cache is implemented on a 16-bit architecture. The cache can hold 8 blocks of memory, each of which is 32 bytes long. How many bits are necessary in order to represent the tag of each of these blocks? Explain how the cache works and how powers of two are used to parse a byte address and find the correct data stored in the cache. (3 points)

c: A dual-core processor system implements a 2-level cache with a per-core L1 cache of size 32Kb and a shared L2 cache of size 128Kb. Explain what operations the caches need to support in order to maintain cahce coherence. What would be the advantages and disadvantages of implementing a write-back or a write-through caching in this system? (3 points)

d: Explain how demand paging works. What are the requirements on the hardware and software side in order to implement a demand paging system? How is the physical memory organized? (3 points)

5 I/O architecture (10 points)

a: What is an interrupt vector and what is it used for? (3 points)

b: When is buffering useful for I/O operations? In what cases can you expect that buffering results in an improved performance and how much of an improvement would a buffer of size K bytes result in compared to using unbuffered operations? (4 points)

c: Explain what are the main advantages of Direct Memory Access (DMA). How can buffer and operation chaining be used to improve DMA performance? (3 points)